Author: Arno Rohner

MySQL with SSL

This Document shows how to configure MySQL for SSL and connect to it with Java

Contents

Last Changes: Luca Niederer, 27.07.2017 (09:45)

Requirements

- MySQL Server
- MySQL Workbench
- Admin Rights

OpenSSL Installation

OpenSSL Download

Download OpenSSL from here.

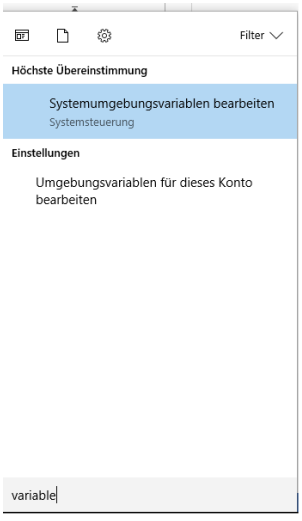
Choose the newest Version (be careful about 32bit/64bit)

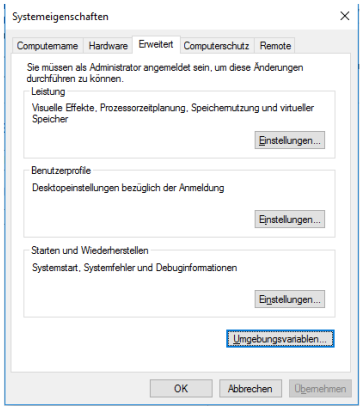
OpenSSL Installation

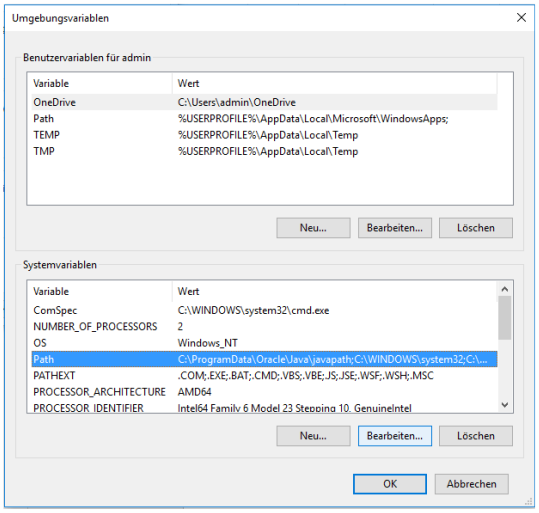
Run through the installer using the default options. **Remember the install directory!**

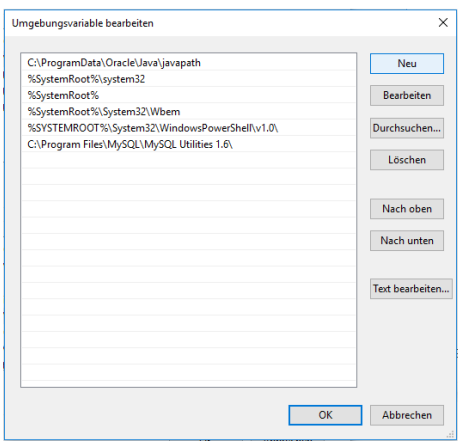
When finished, press the windows button and search for “variable”.
Click on something like “Edit Systemenvoirementvariables”.
Next, click on “Envoirementvariables…” / “Umgebungsvariablen…”

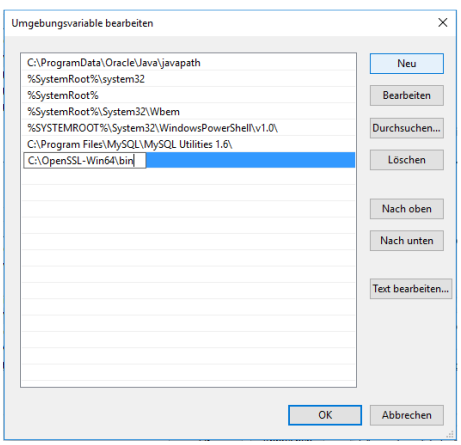
In the bottom half, select “Path” and click “Edit…” / “Bearbeiten…”, then click “New” / “Neu”
In the new Field, enter “*Path\To\OpenSSL*\bin”, for example “C:\OpenSSL-Win64\bin”.









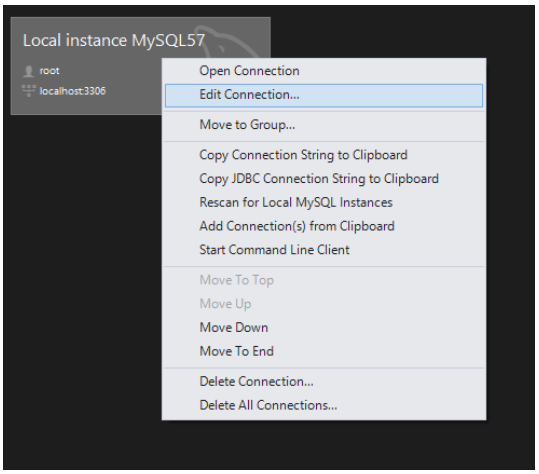


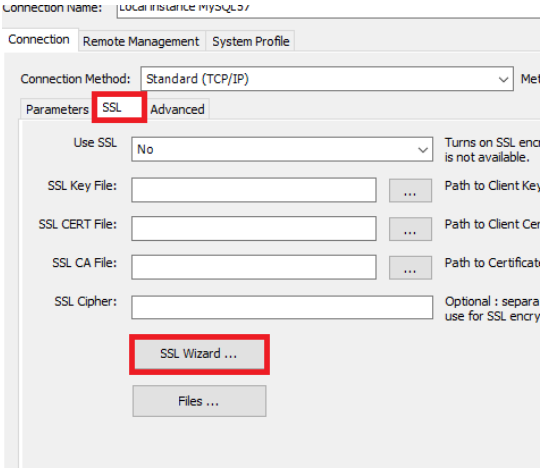
Now test it:
Open the cmd (Windows + R, write “cmd” and press enter).
Enter “openssl”, if it now says “OpenSSL >” the installtion worked, if it tells you that the command could not be found, the installation failed (probably at the point where you set the envoirement variable). In that case just do it again, carefully.

SSL Activation in MySQL

Generate Keys

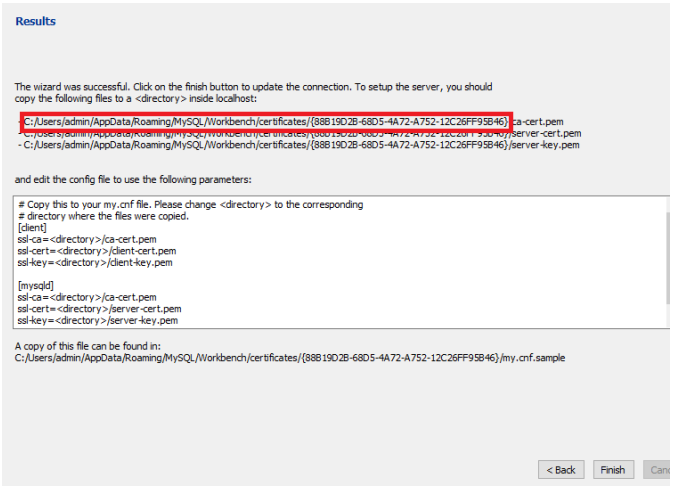
Start the MySQL Workbench as Administrator (just to be safe).
On your Connection you normally use, right click and select “Edit Connection…”.
Go to the SSL Tab and press “SSL Wizard…”.





Click next, **tick “use default parameters”** and click next again.
Copy the path of the files and paste them in the file explorer (go there).

Replace <directory> with the path to the files, so probably “C:\Program Files\MySQL\SSL Keys”.



Open a new File Explorer window and create a new folder called “SSL Keys” in the MySQL install directory (Probably “C:\Program Files\MySQL”).

Copy following files from the generated SSL certificate to the newly created folder:

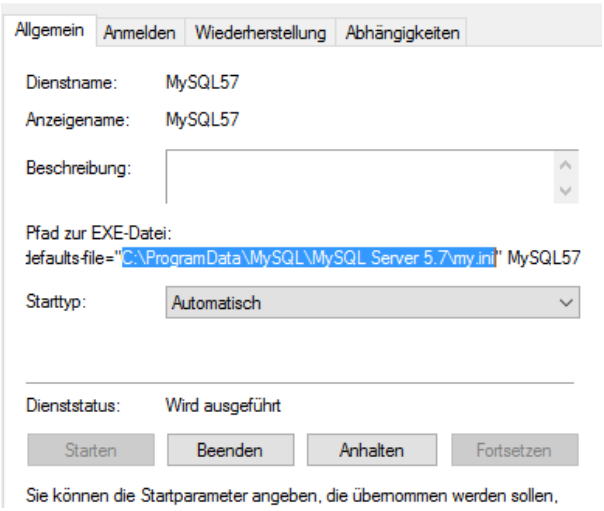
- my.cnf.example
- ca-cert.pem
- client-cert.pem
- client-key.pem
- server-cert.pem
- server-key.pem

Configure Server for SSL

Press the windows key, search for “services” and press enter. Click on an entry and press “m” to jump to the services starting with “m”.

Double click on “MySQL57” (number might change), select the text below “Path to EXE-file:” and drag to the right to see the rest of the text.

Now copy: defaults-file=”copy this text” MySQL57, navigate to that file and open it **as admin** (leave services open, you’ll need it later).



Under [client] paste the client part of the “my.cnf.sample” in “SSL Keys” and under [mysqld] paste the mysqld part. Save the file.

Now go back to services and restart the “MySQL57” service. (Right click -> restart)

Configure Workbench Connection for SSL

Go back to the MySQL Workbench (still in properties of the connection). It should have automatically filled in the SSL Key, CERT and CA file. In the dropdown “Use SSL” select “Require”.

Now press “Test Connection”. The test should succeed.

Connect from Java with SSL

When connecting to the database from Java, you need to change a few properties to use SSL:

```
connectionProperties.put("verifyServerCertificate", "false");
connectionProperties.put("useSSL", "true");
connectionProperties.put("requireSSL", "true");
```

The full code to get a connection object with SSL:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class SSLConnection {

	private static final String MYSQL_DRIVER = "com.mysql.jdbc.Driver";
	private static final String MYSQL_USER = "root";
	private static final String MYSQL_PASSWORD = "admin";
	private static final String MYSQL_HOST = "localhost";
	private static final int MYSQL_PORT = 3306;

	public static Connection getMySQLConnection() throws
ClassNotFoundException, SQLException{
		String dbName = "hwz_test_1";
			Class.forName(MYSQL_DRIVER);
		Properties connectionProperties = new Properties();
			connectionProperties.put("user", "root");
			connectionProperties.put("password", "admin");
			connectionProperties.put("verifyServerCertificate",
"false");
			connectionProperties.put("useSSL", "true");
			connectionProperties.put("requireSSL", "true");
		return DriverManager.getConnection("jdbc:mysql://" + MYSQL_HOST +
":" + MYSQL_PORT + "/" + dbName + "?noAccessToProcedureBodies=true",
connectionProperties);
		}

	public static void main(String[] args){
		try {
			System.out.println(getMySQLConnection());
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}
}
```

"verifyServerCertificate" is set to false, so we don't have to give it a reference to the certificates. Now the certificates sent by the server won't be checked on the client side, but the encryption will still work normal.

Now test the code!